REMARKS

Reconsideration and allowance are respectfully requested.

The specification has been amended to make several editorial changes.

The title has been amended to be more descriptive as required by the Examiner.

The abstract has been amended to include less than 150 words as required by the Examiner.

The Examiner makes several claim objections with suggestions for overcoming those objections. Those suggestions have been implemented.

Claims 2 and 19-21 stand rejected under 35 U.S.C. §112, second paragraph. With regards to the Examiner's objection concerning claim 2, the term "shifted pattern" is described on page 4, lines 18-27 of the application. Claim 2 has been amended to overcome the Examiner's concern. Claim 19 has been amended to clarify that it is a directed to the further step of decompressing the instruction address when it has been encoded and compressed. Claim 21 has been amended as suggested by the Examiner.

Claims 17-21 stand rejected under 35 U.S.C. §101. This rejection is respectfully traversed.

The Examiner contends that these claims are "directed merely to an abstract idea that is not tied to a technological art, environment or machine which would result in a practical application producing a concrete, useful, and tangible result." Applicant disagrees. Claim 17 recites a method of storing in a memory (computer equipment that certainly qualifies as a "technological machine") computer instruction set information. The claim further defines the computer machine by reciting a processing circuit that executes processing instructions from multiple instructions sets stored in memory. Claim 17 has been amended to also recite

"encoding logic" for performing the machine-implemented steps in the method. Since claim 17 (1) is directed to a method that is performed by a computer machine, (2) recites computer structures for carrying out the method (both (1) and (2) preclude the Examiner's mental steps-pencil & paper contention), and (3) produces computer instructions stored in memory that contain information which identify the instruction set to which they belong to allow the processing circuitry to process those instructions properly, claim 17 defines a practical technological invention that produces a concrete, useful, and tangible result.

In addition, claim 21 has been amended to recite a computer readable medium rather than a computer program product as required by the Examiner. Accordingly, claims 17-21 recite statutory subject matter, and the rejection under 35 U.S.C. §101 should be withdrawn.

Claims 1-21 stand rejected under 35 U.S.C. §103 based on the ARM and Nevill references. This rejection is respectfully traversed.

Claim 1 recites that the instruction address has "a predetermined number of bits irrespective of the instruction set to which the associated processing instruction belongs." Claim 1 further recites that while the instruction address has a predetermined number of bits, irrespective of the instruction set to which the associated processing instruction belongs, "a different number of most significant instruction address bits needs to be specified in the instruction address to uniquely identify processing instructions in different instruction sets." The encoding is performed on "at least one" instruction address and is achieved by performing a computation equivalent to "removing any least significant bits not forming the instruction address bits needing to be specified, and extending the specified instruction address bits to n-bits by prepending a pattern of bits to the specified instructions address bits, the number of least significant bits removed and the pattern of bits prepended being dependent on the instruction set

corresponding to that instruction." Similar recitations are recited in the other two independent claims 15 and 17.

The ARM reference describes an Embedded Trace Macrocell (ETM) design developed by the assignee of the present application. In particular, the Examiner refers to section 2.5.1 on page 2-8 of the ARM citation. This ETM design supports both ARM and Thumb instructions. When handling branches, bit zero (the least significant bit) of the address is used to show whether the destination of the branch is ARM code (bit zero low) or Thumb code (bit zero high). This encoding is possible since for both an ARM instruction and a Thumb instruction, bit zero of the address will be a logic zero value and hence can be inferred. Indeed, this property of ARM and Thumb instructions is discussed in the present application at page 13, line 27 to page 14, line 4. In the technique described in the ARM citation, this bit zero property enables an encoding to be provided at the bit zero position. All remaining address bits (i.e. bits 1-31) are unaltered by this technique, and the encoded output in both instances still equals 32 bits.

One problem with the technique disclosed in the ARM citation is that it is only applicable to instruction sets where bit zero of the address will always be zero and thus is free to be used for encoding. But this technique does not have general applicability. For example, it could not be used for the Java instruction set where it is not always the case that bit zero of the address will have a logic zero value. Another problem is that when using the technique described in the ARM citation the output encoded address is always the same size as the original address (i.e. 32 bits in the example given). It is desirable to provide a more efficient encoding.

Claim 1 addresses and overcomes these problems by providing encoding logic which performs a computation equivalent to "removing any least significant bits not forming the

instruction address bits needing to be specified, and extending the specified instruction address bits to n-bits by prepending a pattern of bits to the specified instruction address bits." Removing any least significant bits that can be removed while still enabling the instruction to be uniquely identified, and then prepending a pattern of bits to the remaining specified instruction address bits (i.e. adding the pattern of bits to the opposite end of the address than that from which any appropriate least significant bits were removed), shifts the bit positions of the specified instruction address bits by some predetermined amount.

This instruction address encoding technique is both efficient and flexible. In particular, it is not subject to the restrictions placed on the encoding technique described in the ARM citation (which is only applicable if bit zero of the address is always a logic zero value). Further, the format of the encoded address lends itself to efficient further processing. For example, as defined in claim 4, and as described in the specification with reference to figure 4, such an encoded address lends itself to a particularly efficient compression. This is due in part to the fact that the encoding can produce an n-bit encoded instruction address where a certain number of the most significant bits can be ignored. As contrasted with the technique where the encoded address always contained 32 bits of data which needed to be output, in the preferred example described in the present application, only 31 bits are required if the instruction being encoded is an ARM instruction (since the most significant bit will be zero), which gives a 1-bit savings for each ARM instruction encoded.

With reference to the particular comments raised by the Examiner at the top of page 7 of the Office Action, it will be appreciated from the above description that the Examiner is incorrect in arguing that the encoding logic described in the earlier ARM citation describes a technique

where a bit is **prepended** to the specified instruction address bits, since in accordance with the ARM citation, it is the **least** significant bit and not the most significant bit that is subjected to the encoding. The ARM citation technique *also* neither removes any least significant bits not forming the instruction address bits needing to be specified, nor extends the specified instruction address bits to n-bits by prepending a pattern of bits to the specified instruction address bits.

Nevill describes a technique for incorporating within the program counter register of a data processing apparatus one or more predetermined indicator bits specifying the instruction set. This allows the current instruction set to be changed when a new value is written into the program counter register. That technique improves efficiency when the processor core executes instructions from multiple instruction sets. Nevill discloses two example ways in which the indicator bit could be added to the address in the program counter. See Figures 2 and 3. If the address with the indicator bit added is considered to be the encoded instruction address, then it is clear that this is the instruction address actually used by the processing circuit (the processor core) to address an instruction. But as defined in claim 1, the claimed encoding logic generates an encoded form of instruction address, *which is not used by the processing circuit address an instruction*, but which does contain sufficient information to enable the instruction address to be derived and also provides an indication of the type of instruction set.

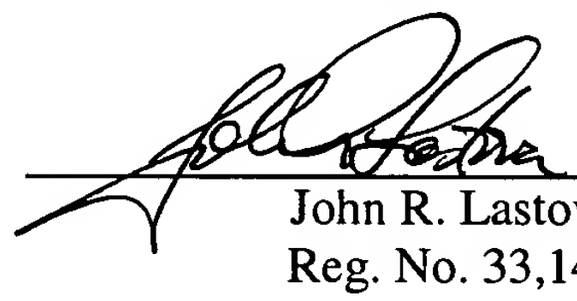
By performing a computation equivalent to removing any least significant bits not forming the instruction address bits needing to be specified, and then extending the remaining specified instruction address bits to n-bits by prepending a pattern of bits to the specified instruction address bits, the bit position of each specified instruction address bit is shifted. No such encoding technique is disclosed in Nevill. In contrast, Nevill's encoded instruction address is formed either by adding the indicator bit as a bit to the left of the most significant bit of the

memory address, in which event the memory address is retained in its original form (see figure 2), or by inserting the indicator bit in the least significant bit position of the memory address if none of the instruction sets make use of that least significant bit (see figure 3). In neither are the specified instruction address bits shifted to different bit positions.

Accordingly, even if the teaching of Nevill is combined with the teaching of the ARM citation, that combination does disclose the encoding logic defined in the independent claims. The application is in condition for allowance. An early notice to that effect is earnestly solicited.

Respectfully submitted,

NIXON & VANDERHYE P.C.

By: 
John R. Lastova
Reg. No. 33,149

JRL:srd
901 North Glebe Road, 11th Floor
Arlington, VA 22203-1808
Telephone: (703) 816-4000
Facsimile: (703) 816-4100